IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Ricard Roma i Dalfo	Examiner:	Mirza, Adnan M.
Serial No.:	09/579,160	Group Art Unit:	2145
Filed:	05/25/2000	Docket No.:	MS149599.1/40062.64US01
Title:	OBJECT-BASED MACHINE AUTOMATION METHOD AND SYSTEM		

CERTIFICATE UNDER 37 CFR 1.10:
"Express Mail" mailing label number: EV 727944334 US
Date of Deposit: February 14, 2006

I hereby certify that this paper or fee is being deposited with the U.S. Postal Service "Express Mail Post Office to Addressee" service under 37 CFR 1.10 on the date indicated above and is addressed to: Mail Stop Appeal Brief-Patents, Commissioner for Patents P.O. Box 1450 Alexandria, Virginia 22313-1450

By: Jenifer Week
Name: Jenifer Week

APPELLANT'S BRIEF ON APPEAL **27488**

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

Sir:

This Brief is presented in support of the Appeal filed November 14, 2005, from the final rejection of claims 1-17, 19-32, and 34-37 of the above-identified application, as set forth in the Office Action mailed October 13, 2005.

A check for $500.00 to cover the required fee for filing this Brief is enclosed.

Also enclosed is a Petition for an Extension of Time and a check to cover the fee for a one-month extension of time.

02/17/2006 MAHMED1 00000024 09579160
01 FC:1402 500.00 OP

I. REAL PARTY OF INTEREST

The real party in interest is Microsoft Corporation.

II. RELATED APPEALS AND INTERFERENCES

None

III. STATUS OF CLAIMS

Claims 1-17 - Appealed

Claim 18 - Canceled

Claims 19-32 - Appealed

Claim 33 - Canceled

Claims 34-37 - Appealed

IV. STATUS OF AMENDMENTS

The Amendment filed on September 7, 2005, subsequent to the Final Office Action in this matter, was entered by the Examiner.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The independent claims generally relate to automating control over one or more client machines, such as personal computers, using a server object of a predefined server object class, a client object of a predefined client object class, and a control module. The control module instantiates the server object in a server process, and the server object instantiates the client object on the client machine.

For purposes of the present appeal, independent claims 1, 16, 34 and 35 stand or fall together and the Board may select any of these claims to decide the appeal with respect to the group. For the purposes of the following argument, reference will be made specifically to the system of claim 1. Independent claim 1 is directed to a machine automation system for automating control of software tests on a client machine under control of a server process. The system includes a predefined machine automation server object class adapted to execute in the server process and a predefined machine automation client object class adapted to execute on the client machine. *See Detailed Description of the Invention*, page 7 line 24-page 8, line 11. The predefined machine automation client object class includes one or more machine automation client objects for executing testing methods on the client machine. *See Id.* at page 7, line 9-23. A machine automation control module instantiates a machine automation server object, of the machine automation server object class in the server process, and instructs the machine automation server object to instantiate one of the machine automation client objects of the machine automation client object class on the client machine to control the testing of the client machine. *See Id.* at page 8, line 21-page 9, line 7.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Applicants appeal the Examiner's rejection of claims 1-17, 19-32, and 34-37 under 35 U.S.C. § 103(a) as being obvious over the combination of U.S. Patent No. 6,199,111 to Hara et al. (hereinafter "Hara") and U.S. Patent No. 6,199,180 to Ote et al. (hereinafter "Ote").

VII. ARGUMENT

The rejection of claims 1-17, 19-32, and 34-37 under 35 U.S.C. § 103(a) as being unpatentable over Hara and Ote is improper and should be reversed by the Board on appeal. As will be discussed in detail below, the references cited by the Examiner fail to teach or suggest each claimed limitation.

A. REJECTION UNDER 35 U.S.C. § 103(A) OVER HARA AND OTE

1. LEGAL AUTHORITY

In order to establish a *prima facie* case of obviousness, the Examiner must establish: 1) some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the references or combine their teachings; 2) a reasonable expectation of success of such a modification or combination; and 3) a teaching or suggestion in the cited prior art of each claimed limitation. *See* MPEP § 706.02(j).

The inherent disclosures of a reference may be used to show that a claim element is present. But "[t]o establish inherency, the extrinsic evidence 'must make clear that the missing descriptive matter is necessarily present in the thing described in the reference, and that it would be so recognized by persons of ordinary skill. Inherency, however, may not be established by probabilities or possibilities. The mere fact that a certain thing may result from a given set of circumstances is not sufficient.'" MPEP § 2112 (quoting *In re Robertson*, 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999) (citations omitted)).

2. THE HARA REFERENCE

Hara "relates to a network computing system for updating and adding data in a distributed client-server system through a network." *Hara*, col. 1, lines 6-8. Under Hara, a client system having a common communication unit communicates via a connection management unit with a plurality of servers each having a common communication unit. *See Id.*, at col. 4, lines 29-65. That is, Hara teaches data communications between a client and one or more servers through a connection management unit. *See Id.* The Examiner concedes that Hara does not teach or suggest (1) a control module instantiating a server object of a server object class; (2) the control module instructing the server object to instantiate a client object of a client object class on the client machine to control operation of the client machine; and (3) the server object instantiating the client object at the instruction of the control module, as required by claim 1. *See Final Office Action of July 27, 2005*, page 2.

3. THE OTE REFERENCE

Ote relates to "a manager for monitoring and controlling faults and performance of a plurality of computers on a network." *Ote*, col. 1, lines 8-15. Under Ote, a service processor board (SVP) – hardware-- in the computer to be managed monitors faults in and controls power to the computer. *See Ote*, col. 3, lines 1-10. A remote management computer communicates with the SVP via an asynchronous communications interface. *See Id.* at col. 4, lines 54-65. Through the connection to the SVP, the remote computer can monitor faults in and control power to the computer to be managed. *See generally Id* at col. 3. Therefore, Ote teaches monitoring of and control of power to a client from a remote management computer through a SVP in the computer to be managed. As stated above, because the Examiner has conceded that Hara does

not disclose specific elements of the claims, the issue on appeal is whether Ote overcomes the deficiencies in Hara.

4. CLAIMS 1-17, 19-32, AND 34-37

Claim 1 is directed to a system with a combination of limitations that includes the following three claim elements: (1) a machine automation control module instantiating a machine automation server object of a machine automation server object class; and (2) the machine automation control module instructing the machine automation server object to instantiate a machine automation client object of a machine automation client object class on the client machine to control the testing of the client machine. As defined on page 8, lines 9-11 of the *Detailed Description of the Invention*, and consistent with common usage, objects are software modules that can encapsulate both data and functionality. Objects are defined by "classes." *See Id.* Additionally, initiation of an object can be accomplished by instantiating the object from a class. *See Detailed Description of the Invention*, page 12 lines 17-23. Instantiation includes creating and initializing the software object.

As stated above, the Examiner concedes that the noted elements of claim 1 are not found within Hara, thus the issue on appeal is whether Ote discloses these three claim elements of claim 1. *See Final Office Action of July 27, 2005*, page 2. As explained below, Ote does not overcomes the deficiencies of Hara.

a. Ote Fails to Disclose A Control Module Instantiating A Server Object

Ote does not teach a software control module that instantiates a software server object, where both the control module and the server object operate in a software server process, i.e., the first noted element of claim 1. The Examiner appears to equate the power controller described in Ote as the control module, and the computer system as the server object. *See Final Office Action July 27, 2005*, page 7. Although the Examiner cites to Hara, the language quoted by the Examiner indicates the Examiner is citing Ote. The power controller and the computer system are not analogous to the control module and the server object, respectively, as recited in claim 1. The specific portion of Ote cited by the Examiner, namely col. 6, lines 24-43 specifically describes the operation of the power controller. As clearly indicated in Ote, the power controller merely controls a power unit to supply power to the managed computer. *See Ote*, col. 6, lines 32-35 and 41-44 ("For the power on request, the power controller 12122 controls power unit 13 to immediately turn on the power.") ("The power controller 12122 receives the power off request and it now immediately controls the power unit 13 to turn off the power.")). The power controller is hardware, and thus cannot instantiate any objects, as the term "instantiate" and "objects," relates to the claim, i.e. software.

Moreover, the computer system disclosed by Ote is not an object instantiated from a class as recited in claim 1. The computer system disclosed in Ote includes several hardware components including a disk and a BUS control circuit. *See Ote*, FIG. 1A and FIG. 5A. It is therefore not an object defined by a class and instantiated by a control module, as required by claim 1. The Examiner is interpreting claim 1 unreasonably in trying to equate hardware

disclosed by Ote with the software limitations of claim 1. A computer system with hardware components is not the same as a server object instantiated from a server class.

Moreover, even alternative readings of Ote cannot reasonably lead to a conclusion that Ote teaches the first noted limitation of claim 1. Ote teaches a remote management computer (*see* Figs. 1A and 1B, numerals 23 and 27) having a manager (*see* Figs. 1A and 1B, numerals 241 and/or 242). *See Ote,* col. 5, lines 9-14. The Examiner may believe that the managers are similar to the control module recited in claim 1. If this conclusion is made, then Ote clearly never teaches the control module instantiating a server object. While Ote describes a fault manager and a power manager (Fig. 1B, numerals 2421 and 2422), Ote does not teach that the manager instantiates the fault manager or the power manager. *See Id.* at col. 5, lines 34-41.

Perhaps the Examiner interprets the SVP manager as the control module. *See Final Office Action July 27, 2005*, page 3. Again, while Ote describes a critical fault manager, a line manager, and a power manager (*see,* Fig. 1B, numerals 293, 291, and 292), Ote does not teach that the SVP manager instantiates the critical fault manager, the line manager, or the power manager. *See Ote* col. 5, lines 56-67. Therefore, even under this alternative interpretation, Ote fails to disclose a control module instantiating a server object, as recited in claim 1.

Based on the disclosure of Ote, the Examiner must be relying on a theory of inherency to conclude that a control module instantiates a server object. However, the Examiner has not presented the required basis in fact or any reasoning whatever to establish that Ote inherently discloses this limitation of claim 1, and for this reason alone, the Board should reverse the rejection.

b. Ote Fails To Disclose A Control Module Instructing A Server Object To Instantiate A Client Object On A Client Machine

Further, with respect to the second noted limitation of claim 1, it is not clear what features of Ote the Examiner is equating to the client object. On Page 8 of the *Final Office Action*, the tries to point to features of Ote that apparently the Examiner believes equates to the control module, the server object and the client object. Specifically, the Examiner refers to several features of Ote, including the preset power time controller, the SVP driver, the agent, the network OS, the power controller, and the SVP board. *See Final Office Action of July 27, 2005*, page 8. These features cited by the Examiner are all on a single machine, i.e., the managed computer. *See Ote*, FIG. 1A and FIG. 5A. However, claim 1 clearly requires that the server object and control module operate in the server process, while the client object is on the client machine. Thus, regardless of how the Examiner is interpreting those features in Ote to correspond to the limitations of claim 1, they cannot meet, the machine automation control module instructing the machine automation server object to instantiate a machine automation client object of a machine automation client object class on the client machine to control the testing of the client machine.

Unfortunately, the Examiner's conclusions cannot be supported by any interpretation of Ote. For example, the Examiner may equate the agent disclosed by Ote to the client object of claim 1. Ote teaches that the manager connects to an agent on the managed computer. *See Ote*, col. 5, lines 9-14 and col. 5, lines 35-41. However, Ote does not teach that the manager, the fault manager, or the power manager instantiates the agent. Nor does Ote have any additional

disclosure indicating that the agent is instantiated by a server object. Thus, Ote does not teach a server object instantiating an object of a client automation class as defined in claim 1.

Using an alternative interpretation, the power controller disclosed by Ote could be equated by the Examiner to the client object in claim 1. Ote discloses that the power controller is connected to the SVP manager. *See Ote,* col. 5, lines 63-66. However, the SVP manager never instantiates the power controller. *See, Id.* at col. 5, line 62 – col. 6, line 5. While Ote is replete with instances where the power controller sends commands to and receives commands from various hardware systems (*see* col. 5, lines 62 – col. 6, line 44; col. 8, lines 19-21), there is not a single mention, in Ote, of the power controller being instantiated by something akin to the machine automation server object. The SVP manager may communicate with the power controller, but the SVP manager does not instantiate the power controller. Indeed, the power controller is always powered (*see* col. 4, lines 59-60), and thus, would never need to be instantiated or re-instantiated by a server object, even if it could be compared to the client object.

c. Conclusion

Accordingly, the Examiner has not established a prima facie case of obviousness, because the references cited by the Examiner fail to disclose all the limitations of claim 1. Specifically, the Examiner has not pointed to features disclosed by Ote that show the following elements of claim 1: (1) a machine automation control module instantiating a machine automation server object of a machine automation server object class; and (2) the machine automation control module instructing the machine automation server object to instantiate a machine automation client object of a machine automation client object class on the client machine to control the

testing of the client machine. Nor has the Examiner presented the required basis in fact or reasoning to establish that Ote inherently discloses the noted limitations of claim 1. Therefore, Applicants respectfully request that the Board reverse the Examiner's rejection of claims 1-17, 19-32 and 34-37 under 35 U.S.C. § 103(a) as being obvious over the combination of Hara and Ote.

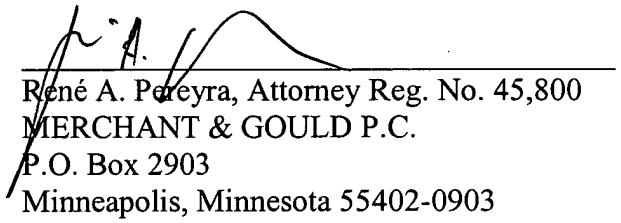
VIII. SUMMARY

It is earnestly requested that the Examiner's rejection be reversed, and that all of the pending claims be allowed.

Please charge any additional fees or credit overpayment to Merchant & Gould Deposit Account No. 13-2725.

Respectfully submitted,

Date: February 14, 2006



René A. Pereyra, Attorney Reg. No. 45,800
MERCHANT & GOULD P.C.
P.O. Box 2903
Minneapolis, Minnesota 55402-0903
(612) 332-5300

27488

IX. CLAIMS APPENDIX

1. (Previously Presented) A machine automation system for automating control of software tests on a client machine under control of a server process, the system comprising:

a predefined machine automation server object class adapted to execute in the server process;

a predefined machine automation client object class adapted to execute on the client machine in communication with an instance of the machine automation server object class, and the predefined machine automation client object class comprises one or more machine automation client objects for executing testing methods on the client machine; and

a machine automation control module instantiating a machine automation server object of the machine automation server object class in the server process and instructing the machine automation server object to instantiate one of the machine automation client objects of the machine automation client object class on the client machine to control the testing of the client machine.

2. (Previously Presented) The machine automation system of claim 1 further comprising:

a machine identifier identifying the client machine on which the machine automation client object is to be instantiated; and

a machine identifier source providing the machine identifier to the machine automation server object to initiate instantiation of the machine automation client object on the client machine.

3. (Original) The machine automation system of claim 1 wherein the machine automation server object includes a shutdown server object executing in the server process of a server machine, the machine automation client object includes a shutdown client object executing in a client process of the client machine, and the machine automation control module instructs the shutdown server object to cause the shutdown client object to reboot the client machine and to re-establish communications with the shutdown client object via a communications mechanism after rebooting of the client machine completes.

4. (Original) The machine automation system of claim 3 wherein the machine automation control module continues execution after rebooting of the client machine completes.

5. (Original) The machine automation system of claim 4 further comprising:

a timeout field of the machine automation server object for storing a timeout value specifying an amount of time the machine automation server object waits to return control to the machine automation control module after rebooting of the client machine completes.

6. (Original) The machine automation system of claim 3 wherein the machine automation control module continues execution after the shutdown server object causes the shutdown client object to reboot the client machine and before rebooting of the client machine completes.

7. (Original) The machine automation system of claim 1 wherein the machine automation server object includes a shutdown server object executing in the server process of a server machine, the machine automation client object includes a shutdown client object executing in a client process of the client machine, and the machine automation control module

instructs the shutdown server object to cause the shutdown client object to log off of the client machine and to re-log in to the client machine using a predetermined user name without rebooting.

8. (Original) The machine automation system of claim 7 wherein the predetermined user name is recorded in a system registry of the client machine and read from the system registry by the machine automation client object to re-log in to the client machine.

9. (Original) The machine automation system of claim 1 wherein the machine automation server object includes a shutdown server object executing in the server process of a server machine, the machine automation client object includes a shutdown client object executing in a client process of the client machine, and the machine automation control module instructs the shutdown server object to cause the shutdown client object to reboot the client machine and to re-log in to the client machine after rebooting completes.

10. (Original) The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine automation client object to access a system registry of the client machine.

11. (Original) The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine automation client object to return system information about the client machine to the machine automation server object.

12. (Original) The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine

automation client object to return to the machine automation server object a snapshot of a system registry of the client machine.

13. (Original) The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine automation client object to return to the machine automation server object a snapshot of a file system of the client machine. The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine automation client object to install an application on the client machine.

14. (Original) The machine automation system of claim 1 wherein the machine automation control module instructs the machine automation server object to cause the machine automation client object to install an application on the client machine.

15. (Previously Presented) The machine automation system of claim 1 further comprising:

a first machine identifier received by the machine automation server object identifying the client machine;

another machine automation server object adapted to execute in the server process;

a second machine identifier received by said another machine automation server object identifying another client machine; and

another machine automation client object identified by the second machine identifier and adapted to execute on said another client machine in communication with said another machine automation server object via a communications mechanism.

16. (Previously Presented) A method for automating control of a client machine under control of a server process, the method comprising:

executing a machine automation control module in the server process;

instantiating a machine automation server object of a predefined machine automation server object class in the server process, under command of the machine automation control module;

instructing the machine automation server object to instantiate a machine automation client object of a predefined machine automation client object class on the client machine;

instructing the machine automation server object to cause the machine automation client object to reboot the client machine; and

re-establishing communications between the machine automation server object and the machine automation client object via a communications mechanism after rebooting of the client machine completes.

17. (Previously Presented) The method of claim 16 further comprising:

providing the machine automation server object with an identifier of the client machine on which the machine automation client object is to be instantiated; and

instantiating the machine automation client object on the client machine specified by the identifier.

19. (Previously Presented) The method of claim 16 further comprising:

restoring a value of a property of the machine automation client object after rebooting of the client machine completes.

20. (Previously Presented) The method of claim 16 further comprising:

executing a first instruction of the machine automation control module that instructs the machine automation server object to cause the machine automation client object to reboot the client machine; and

delaying execution of a subsequent instruction of the machine automation control module until after rebooting of the client machine completes.

21. (Previously Presented) The method of claim 20 wherein the delaying operation delays execution of a subsequent instruction of the machine automation control module until after a specified time after rebooting of the client machine completes.

22. (Previously Presented) The method of 16 further comprising:

executing a first instruction of the machine automation control module that instructs the machine automation server object to cause the machine automation client object to reboot the client machine; and

executing a subsequent instruction of the machine automation control module after the machine automation server object causes the machine automation client object to reboot the client machine and before rebooting of the client machine completes.

23. (Previously Presented) The method of claim 16 further comprising:

executing the server process in a server machine;

instructing the machine automation server object to cause the machine automation client object to log off of the client machine; and

instructing the machine automation server object to cause the machine automation client object to re-log in to the client machine using a predetermined user name.

24. (Previously Presented) The method of claim 23 further comprising:

recording the predetermined user name in a system registry of the client machine; and

instructing the machine automation server object to cause the machine automation client object to log into the client machine using the predetermined user name read from the system registry.

25. (Previously Presented) The method of 16 further comprising:

executing a first instruction of the machine automation control module that instructs the machine automation server object to cause the machine automation client object to reboot the client machine; and

executing a subsequent instruction of the machine automation control module after the machine automation server object causes the machine automation client object to reboot the client machine and before rebooting of the client machine completes.

26. (Original) The method of claim 16 further comprising:

executing the server process in a server machine;

instructing the machine automation server object to cause the machine automation client object to reboot the client machine; and

instructing the machine automation server object to cause the machine automation client object to re-log in to the client machine after rebooting completes.

27. (Original) The method of claim 16 further comprising:

instructing the machine automation server object to cause the machine automation client object to access a system registry of the client machine, under command of the machine automation control module.

28. (Original) The method of claim 27 wherein the operation of instructing the machine automation server to cause the machine automation client object to access the system registry comprises:

instructing the machine automation server object to cause the machine automation client object to return to the machine automation server object a snapshot of the system registry of the client machine, under command of the machine automation control module.

29. (Original) The method of claim 16 further comprising:

instructing the machine automation server object to cause the machine automation client object to return to the machine automation server object a snapshot of a file system of the client machine, under command of the machine automation control module.

30. (Original) The method of claim 16 further comprising:

instructing the machine automation server object to cause the machine automation client object to return system information about the client machine to the machine automation server object, under command of the machine automation control module.

31. (Original) The method of claim 16 further comprising:

instructing the machine automation server object to cause the machine automation client object to install an application on the client machine, under command of the machine automation control module.

32. (Previously Presented) The method of claim 16 further comprising:

providing to the machine automation server object a first machine identifier identifying the client machine;

instantiating another machine automation server object of the machine automation server object class in the server process, under command of the machine automation control module;

providing to said another machine automation server object a second machine identifier identifying another client machine;

instructing said another machine automation server object to instantiate another machine automation client object of the machine automation client object class on said another client machine via a communications mechanism; and

instructing said another machine automation server object to cause said another machine automation client object to control operation of said another client machine.

34. (Previously Presented) A computer program storage medium readable by a computer system and encoding a computer program for executing a computer process automating control of a client machine under control of a server process, the computer process comprising:

executing a machine automation control module in the server process;

instantiating a machine automation server object of a predefined machine automation server object class in the server process, under command of the machine automation control module;

instructing the machine automation server object to instantiate a machine automation client object of a predefined machine automation client object class on the client machine;

instructing the machine automation server object to cause the machine automation client object to reboot the client machine; and

re-establishing communications between the machine automation server object and the machine automation client object via a communications mechanism after rebooting of the client machine completes.

35. (Previously Presented) A computer program storage medium readable by a computer system and encoding a computer program for executing on a computer system a computer process for automating control of a first client machine and a second client machine under control of a server process via a communications mechanism, for testing software on the first client machine and the second client machine, the computer process comprising:

executing a machine automation control module in the server process;

instantiating a first machine automation server object of a predefined machine automation server object class in the server process, under command of the machine automation control module;

instructing the first machine automation server object to instantiate a first machine automation client object of a predefined machine automation client class on the first client machine, and the first machine automation client object for executing one or more testing methods on the client machine;

instantiating a second machine automation server object of the machine automation server object class in the server process, under command of the machine automation control module;

instructing the second machine automation server object to instantiate a second machine automation client object of the machine automation client object class on the second client machine, and the second machine automation client object for executing one or more testing methods on the client machine;

instructing the first machine automation server object to cause the first machine automation client object to control testing of the first client machine; and

instructing the second machine automation server object to cause the second machine automation client object to control testing of the second client machine.

36. (Previously Presented) The computer program product of claim 35 wherein the computer process further comprises:

executing the server process in a server machine; and

instructing the first machine automation server object to cause the first machine automation client object to reboot the first client machine;

instructing the second machine automation server object to cause the second machine automation client object to reboot the second client machine;

re-establishing communications between the first machine automation server object and the first machine automation client object via the communications mechanism after rebooting of the first client machine completes; and

re-establishing communications between the second machine automation server object and the second machine automation client object via the communications mechanism after rebooting of the second client machine completes.

37. (Original) The computer program product of claim 36 wherein the computer process further comprises:

rebooting the first client machine, responsive to the operation of instructing the first machine automation server object to cause the first machine automation client object to reboot the first client machine; and

rebooting the second client machine concurrently with the operation of rebooting the first client machine, responsive to the operation of instructing the second machine automation server object to cause the second machine automation client object to reboot the second client machine.

X. EVIDENCE APPENDIX

A. OFFICE ACTIONS AND AMENDMENTS/RESPONSES

1. Advisory Action -- mailed October 13, 2005
2. Amendment under Rule 116 -- mailed September 7, 2005
3. Final Office Action -- mailed July 27, 2005
4. Amendment -- mailed December 6, 2004
5. Office Action -- mailed September 8, 2004

B. REFERENCES RELIED UPON BY THE EXAMINER

1. U.S. Patent No. 6,199,111 to Hara et. al ("Hara")
2. U.S. Patent No. 6,199,180 to Ote et. al ("Ote")

C. REFERENCES CITED BY APPELLANTS

1. Drive Image - Complete Data Protection and Backup Solution "Driveimage Information", © 2000 PowerQuest Corporation
2. Norton Ghost Enterprise Edition "Norton Ghost 6.0 Enterprise Edition Product Info" © 1995-2000 Symantec Corporation
3. IEEE Internet Computing, Vol. 2, No. 1, January/February 1998 "Virtual Network Computing", The Olivetti & Oracle Research Laboratory

4. Virtual Network Computing "VNC documentation", AT&T Laboratories Cambridge 1999

5. Virtual Network Computing "What is VNC? - A practical introduction, AT&T Laboratories Cambridge

D. CASES CITED IN THE BRIEF

In re Robertson, 169 F.3d 743, 745, 49 USPQ2d 1949, 1950-51 (Fed. Cir. 1999)

XI. RELATED PROCEEDINGS APPENDIX

None